**Connecting the Dots Between News Articles**

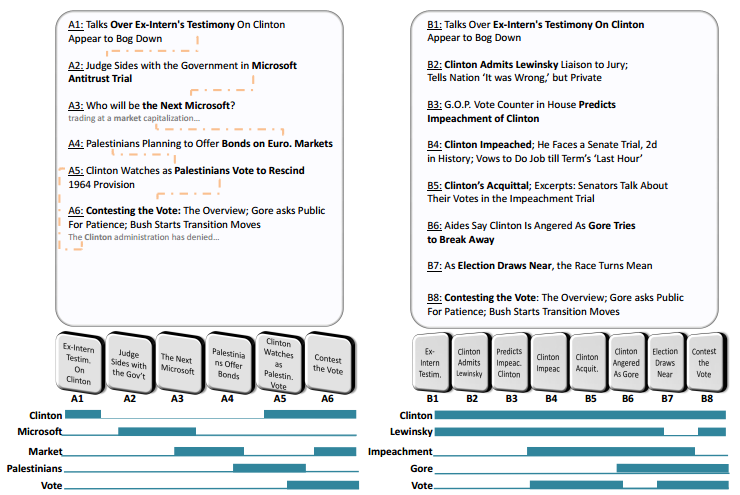
**Dafna Shahaf, Carlos Guestrin**

The aim of the author is to build a chain of documents to connect two documents which were published at different times. The input to the algorithm is a set of documents with their timestamps, a start document(*s*) and an end document(*t*). The given set of documents can be chronologically ordered using their timestamps. Let there be *n* documents between *s* and *t.* There are 2n possible chains between s and t. There is need of scoring mechanism to choose the best possible chain.

Author has proposed two metrics for scoring a chain – *coherency* and *influence*. A coherent chain has a global coherent theme across the storyline i.e. all the documents in the chain belong to a single theme. Two consecutive documents in a chain has high influence(for a given word w) if the two documents are highly connected and w is important for connectivity.

The steps involved in the algorithm are as follows –

An example of coherent chain vs incoherent chain is given below.



In the example, the left chain is erratic and the documents are not logically connected. The right chain has highly connected documents and thus is more coherent.

Let D be a set of articles, and W a set of features (typically words or phrases). Each article is a subset of W. Given a chain (d1, ..., dn) of articles from D. An intuitive way to form a coherent chain is that every time a word appears in two consecutive documents we score a point.



Thus similar documents are placed next to each other. But it has 4 drawbacks – weak links, missing words, importance, jitteriness.

**Weak links:** A chain can have high coherence is most of the links are strong while few links are weak. Summing over the transitions can lead to ‘broken’ chains (having weak links). A more reasonable way is to consider the minimal transition score instead of the sum.



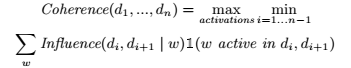
**Missing Links**: There are cases in which some words do not appear in an article, although they should have. For example, if a document contains ‘lawyer’ and ‘court’ but not ‘prosecution’, chances are ‘prosecution’ is still a highly-relevant word. Considering only words from the article can be misleading in such cases.

**Importance:** Some words are more important than others, both on a corpus level and on a document level. Two documents can have numerous common words but more important words must have influence on the transition between the two documents.

To address above two problems, the author suggests the concept of influence of di on di+1 through the word w. The calculation of Influence(di, di+1 | w) has been discussed later. Intuitively, Influence(di , dj | w) is high if (1) the two documents are highly connected, and (2) w is important for the connectivity.



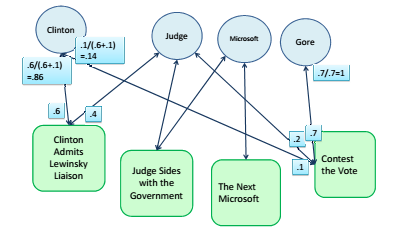
**Jitteriness:** Jitteriness is appearance and disappearance of patterns throughout the chain.

One way to avoid jitteriness is to consider the longest continuous stretch of each word. But words can have high influence on a transition even if they do not appear in the documents. The author defines an activation pattern arbitrarily for each word, and compute the objective based on it. The coherence is then defined as the score under the best activation pattern.  


Word acitvations can be binary or continuous. The author has considered word activation in the range [0,1]. The above mentioned equation is called ‘**Objective function**’.

**Calculating *Influence(di, dj | w)***

Construct a bipartitie graph, G = (V,E), where V = VD U VW , VD is set of documents, VW is set of words. Edge weights represent the strength of the correlation between a document and a word. Author has used tool www.copernic.com to assign importance to each word and use these weights for document-to-word edges. Alternatively, TF-IDF weights can also be used. Since weights are interpreted as random walk probabilities, they are normalized over all words in the document.



Intuitively, if the two documents are connected, a short random walk starting from di should reach dj frequently. The stationary distribution is the fraction of the time the walker spends on each node:



Where Πi (v) is the stationary distribution of random walk starting from di. P (v | u) is the probability of reaching v from u and ϵ is random restart probability.

Let Πwi (v) be the stationary distribution for graph which has as a sink node. If w was influential, the stationary distribution of dj would decrease a lot The influence on dj w.r.t. w it defined as the difference between these two distributions, **Πi (dj ) – Πwi (dj ).**

**Generating Event Storylines from Microblogs**

Chen Lin , Chun Lin , Jingxuan Li , Dingding Wang , Yang Chen , Tao Li

Review submitted by *Anunaya Srivastava*

**AIM**

To generate storylines from microblogs(Twitter) for user input queries.

**INPUT**

1. Tweet dataset
2. User query

NB: Start point and end point for the storyline is not given.

**NOVELTY**

* Storyline is generated from Twitter data which has fixed length, dynamic characteristics, retweets for the same event. It also has similar/duplicate tweets in large numbers.
* This work focuses on event detection with a global view i.e. it also covers previous documents also, while the previous work considers only recent documents.
* This work adds a temporal dimension to the storyline i.e. the storyline generated also has temporal data stored with it.
* It is dynamic i.e. it selects event specific expansion terms, which are temporally correlated with query terms in pseudo relevant documents.

**HIGH LEVEL APPROACH**

**DETAILED APPROACH**

1. Relevant tweet retrieval based on user query

* To enhance the query expressibility, query expansion is adopted to replace the original query Q by a new high quality query Q’.
* In a pseudo relevance manner, suppose the few top ranked documents d+ by the initial query Q builds a relevant model θF, we can set the new query to be a linear combination of original query Q and relevant model θF



* Relevance model method is followed to infer θF



**Dynamic Pseudo Relevance Feedback**

* In traditional pseudo relevance feedback (PRF), the prior p(d+) is usually set to be uniform. However, this assumption doesn’t hold in an instant broadcast medium like Twitter.
* Intuitively, in the initial search results of an event query of “Egypt Revolution”, a top tweet published on 2011-01-25 is more likely to be a truly relevant tweet than a tweet published on 2011-01-01 on a near position in the ranking list. Suppose that the event is detected to have K burst periods (detection detail is introduced in the next subsection), the prior distribution of relevant tweets should be centered around each burst period.
* DRPF is dynamic i.e. prior probability of relevant document is given by a probability distribution. Different prior probability distribution can be used. This paper has considered the following 3 distributions –
  1. Mixture Gaussian Distribution: assumes that the prior probability is normally distributed, with multiple peaks located at the centroid of each burst period.
  2. Local Power Distribution: assumes that the prior probability is restricted to the neighborhood around the nearest burst period.
  3. Skewed Linear Distribution: assumes that the prior probability is positive skew, with a longer tail after the burst period.
* DRPF gives a global view of tweets and not just recent tweets.

1. Summarization

* This paper has compared the following different summarization techniques. The last technique(Dominant Set) is proposed by the author and he finds it to be the most efficient technique.
  1. Random: randomly selects the sentence as the summary
  2. Most Relevant: picks up the sentences which are most relevant with the topic as the summary
  3. Latent Semantic Analysis (LSA): identifies semantically important sentences by conducting latent semantic analysis
  4. K-means: performs K-means over the sentences, then treats centers of all sentence clusters as the summary
  5. Non-negative Matrix Factorization (NMF): performs NMF on the sentence-term matrix and selects the high ranked sentences
  6. Symmetric Non-negative Matrix Factorization (SNMF): calculates sentence-sentence similarities by sentence level semantic analysis, clusters the sentences via symmetric non-negative matrix factorization, and extracts the sentences based on the clustering result
  7. Spectral Clustering with Normalized Cuts (NCut): performs the Spectral Clustering using Normalized Cut to cluster the sentences, and then uses centers of clusters as the summary
  8. Query-sensitive Mutual Reinforcement Chain (Qs-MRC): extends the mutual reinforcement principle between sentence and term to document-sentence-term mutual reinforcement chain, and uses query-sensitive similarity to measure the affinity between the pair of texts
  9. Multi-Document Summarization using Submodularity (MSSF): a multi-document summarization framework based on Submodularity
  10. Dominant Set (DS only)[3]: Document summarization using the Dominant Set algorithm.
* The summarization approach followed is –

**MULTI-VIEW TWEET GRAPH**

A multi-view graph G=(V,W,E,A), where V is a set of vertices (nodes), W is the weights of V, E is a set of undirected edges, which represents the similarities between tweets, and A is a set of directed edges (arcs), which represents the time continuity of the tweets.

Construction of such a graph is controlled by three nonnegative real parameters α, τ1,τ2, τ1<τ2

Vi -> Vj iff

tweet similarity is greater than α

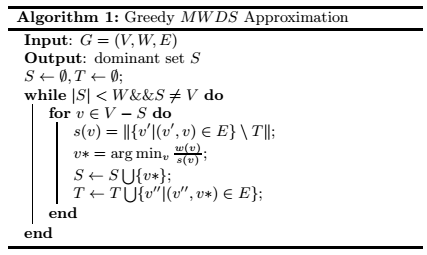
τ1 <= tj – ti <= τ2

score(Q,Vi) is calculate using cosine similarity.

Vertex weight, w(Vi) = 1 – score(Vi)

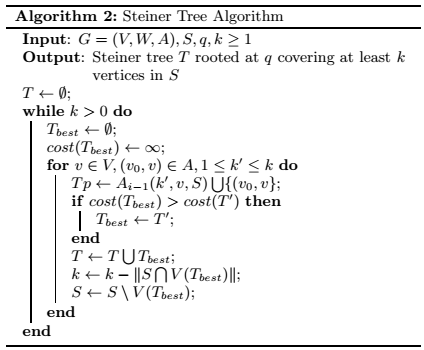
**MDWS Algorithm**

Input: Graph G=(V,W,E) where V is the set of vertices each representing



1. Storyline Generation

* Steiner-tree algorithm is used for storyline generation.



* Steiner-tree algorithm can detect the outline of all given sentences from dominant sets.